SISTEM MANAJEMEN DAN VISUALISASI *SYSLOG* PERANGKAT JARINGAN KOMPUTER PADA ICT UNIVERSITAS DIPONEGORO BERBASIS ELK STACK

Adnan Fauzi
Departemen Teknik Komputer, Fakultas Teknik, Universitas Dipongoro
Jl. Prof. Soedarto, SH, Kampus Undip Tembalang, Semarang, Indonesia 50275
adnan@live.undip.ac.id

Abstract—*Device monitoring is an important thing to manage networks. Information related network state or condition can be gathered through the device monitoring for administrators to make decisions regarding occurred events. Logs can be useful information to monitor network devices. Network administrator of Diponegoro University need a centralized log management system that can receive, manage, and analyze logs. This research identify functional requirements of log management system. Topology design and software that will be used by the log management system use DSR method. The next step is implementation of the topology, software, and application. The last step is testing the system and log management application. The results show that collecting centralized logs and processing these logs into information in the form of dashboards using ELK Stack application successfully implemented. The dashboard resulted by ELK Stack Application will be implemented on the web application using PHP programming language and Codeigniter framework. The test results show that system can receive logs and group the log according to the device location and the severity level of the log.*

Keywords — *Monitoring; log; ELK Stack; PHP; Codeigniter; Dashboard*

I. PENDAHULUAN

Perkembangan teknologi informasi saat ini mampu menghasilkan beraneka ragam layanan-layanan baru yang bermanfaat membantu pekerjaan manusia. Kenanekaragaman teknologi tersebut memberikan kemudahan bagi para pengguna teknologi dalam implementasi penyebaran informasi [1]. Kemudahan penyebaran informasi yang dirasakan saat ini tidak bisa terlepas dari peran administrator jaringan dalam mengelola sumber daya jaringan dengan baik.

Jaringan komputer adalah himpunan interkoneksi antara 2 komputer *autonomous* atau lebih yang terhubung dengan media transmisi kabel atau tanpa kabel [2]. Administrator jaringan bertanggung jawab mendesain, mengkonfigurasi perangkat jaringan, serta menjaga stabilitas *traffic*. Perangkat jaringan, selaku objek yang dikelola administrator tidak jarang mengalami kendala atau masalah. Masing-masing masalah memiliki penyebab yang beragam, bisa dikarenakan kesalahan administrator, tidak adanya dokumentasi, atau eksploitasi perangkat oleh pihak yang tidak berwenang. Masing-masing masalah menjadi sebuah *event* dari perangkat jaringan yang semua *event* tersebut terekam dalam sistem *log* (*syslog*).

Syslog adalah cara bagi perangkat jaringan untuk mengirim *event message* ke *server logging* atau yang biasanya dikenal sebagai *server sys*log. Memahami *syslog* merupakan cara paling efektif untuk "mendengarkan" pesan *server* kepada pengguna [3]. Masalah yang muncul berikutnya adalah, pada jaringan yang besar dan terdapat banyak perangkat akan menyulitkan administrator jaringan untuk mendeteksi sumber dan penyebab kendala di perangkat jaringan. Sehingga diperlukan suatu sistem manajemen *log* terpusat yang dapat memusatkan pengiriman *log* dari seluruh perangkat jaringan serta memiliki sistem pelaporan yang mudah dianalisa oleh administrator jaringan.

Penelitian [4] memaparkan implementasi sistem manajemen *log* yang menggunakan aplikasi Splunk, yaitu aplikasi komersial yang didesain khusus untuk manajemen *log* terpusat. Hasil dari penelitian ini adalah desain sistem manajemen *log* menggunakan *query* dan alur kerja dari Splunk, yaitu *Filtering, Reformatting,* dan *Summarization.*

Penelitian [5] memaparkan perancangan dan strategi implementasi manajemen *log* di lingkungan jaringan WAN. Hasil dari penelitian ini adalah, terdapat beberapa tahapan perancangan sistem manajemen *log* di jaringan WAN, yaitu mengurutkan kebutuhan, inventaris aset, memeriksa topologi jaringan, memilih *log* yang akan digunakan, memilih arsitektur infrastruktur, pembuatan *log*, pengumpulan *log*, sinkronisasi waktu, pemrosesan *log*, mengawasi skalabilitas, dan pengukuran performa.

ELK Stack atau gabungan dari aplikasi *opensource* Elasticsearch, Logstash dan Kibana adalah *tool* yang berguna untuk mengumpulkan *log* dan juga menvisualisasi *log* tersebut, Elasticsearch berguna untuk menyimpan semua *log*, Logstash digunakan untuk mengumpulkan dan mem-*parsing log* kemudian disimpan pada Elasticsearch. Kibana adalah *web interface* yang berguna untuk menampilkan *log* baik dalam bentuk grafik maupun visualisasi lainnya [6].

Melalui penelitian ini, penulis bermaksud mengimplementasikan sistem manajemen dan visualisasi *syslog* perangkat jaringan komputer berbasis ELK Stack yang dibuat untuk mengintegrasikan *log* dari perangkat jaringan yang digunakan ICT Universitas Diponegoro dan memvisualisasikan data dari perangkat tersebut guna memudahkan administrator jaringan untuk menganalisa dan mengambil tindakan dari masalah pada perangkat jaringan yang dikelola.

II. METODE PENELITIAN

Metodologi penelitian yang digunakan untuk merancang bangun sistem manajemen *log* adalah metodologi *Design Science Research* (DSR), tahapan-tahapan dalam metodologi DSR adalah definisi sistem, spesifikasi sistem, konfigurasi sistem, evaluasi, dan hasil.

Tahap pertama yang dilakukan adalah definisi sistem. Tahap ini mendefinisikan sistem yang akan dibuat, meliputi penjabaran sistem, identifikasi kebutuhan dan manfaat sistem.

Tahap selanjutnya yang dilakukan adalah spesifikasi sistem. Proses spesifikasi kebutuhan akan menjabarkan tentang awal perancangan sistem dengan menentukan spesifikasi kebutuhan yang sesuai definisi sistem.

Tahap ketiga adalah konfigurasi sistem. Pada tahap ini, spesifikasi kebutuhan yang telah ditentukan akan dirancang sesuai topologi/desain jaringan dan direalisasikan sebagai

serangkaian sistem atau unit sistem yang memungkinkan untuk menjalankan tujuan sistem dan cara kerja sistem.

Tahap keempat adalah evaluasi. Pengujian dilakukan mulai tanggal 29 April 2019 sampai 8 Juli 2019 menghasilkan 2.354.224 *log* yang berhasil terkumpul di *server* yang terdiri dari 1.085.368 (46,1%) pesan warning, 569.892 (24,21%) pesan notification, 450.434 (19,133%) pesan informational, 248.266 (10,544%) pesan error, 262 (0.01292%) pesan critical, dan dua (0.000085%) pesan alert. Tahap terakhir adalah hasil. Penelitian dianggap berhasil jika sistem yang dibuat telah memenuhi tujuan dari penelitian.

A. Analisis Kebutuhan Fungsional

Kebutuhan fungsional dari sistem manajemen *log* yaitu sistem dapat mengumpulkan *log* aktifitas dari perangkat jaringan secara *remote* dengan menggunakan *syslog*. Sistem dapat mengelompokkan *log* berdasarkan tingkat *severity* pada *log* tersebut. *Severity level* merepresentasikan tingkat urgensi dari suatu pesan *log*. *Severity level* 0 (*Emergency*) berisi pesan dengan urgensi paling tinggi dan *severity level* 6 (*Informational*) berisi pesan dengan urgensi paling rendah. Kebutuhan fungsional yang terakhir yaitu Sistem dapat memvisualisasikan *log* menjadi informasi yang mudah dipahami oleh administrator jaringan.

B. Analisis Kebutuhan Perangkat Keras

Pada penelitian ini perangkat keras digunakan sebagai wadah sistem dan penunjang agar sistem dapat berjalan. Sistem diimplementasikan pada sebuah *virtual machine*. *Virtual machine* yang digunakan memiliki spesifikasi pada Tabel 1

.

Tabel 1. Spesifikasi *Virtual Machine*

Komponen	Server
Prosesor	Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz
Motherboard	Intel 440BX
RAM	4 GB
HDD	60 GB
NIC	1 *Fast Ethernet*

1 Unit Switch Cisco WS-C4510R+E di ICT sebagai perangkat utama yang menghubungkan seluruh perangkat jaringan di Universitas Diponegoro. Router Cisco 2921 dan switch Cisco WS-C2960X-24TD-L sebagai perangkat jaringan untuk dilakukan *monitoring* pada tiap fakultas dan unit di Universitas Diponegoro. Satu laptop Toshiba Portege R30 difungsikan untuk melakukan konfigurasi sistem dan membuat aplikasi.

C. Kebutuhan Perangkat Lunak

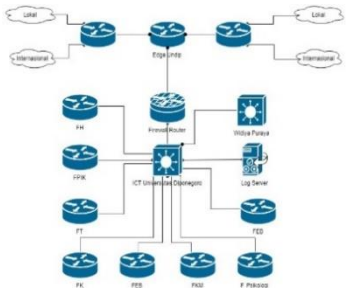
Sistem manajemen *log* membutuhkan perangkat lunak untuk membangun *server* dan komponen pendukung lainnya. *Server* manajemen *log* menggunakan sistem operasi Ubuntu 18.04 LTS. Aplikasi untuk melakukan pengumpulan *log* menggunakan Logstash untuk kemudian disimpan pada Elasticsearch dan divisualisasikan menggunakan Kibana [7].

Pembuatan aplikasi monitoring menggunakan bahasa pemrograman PHP dengan kerangka kerja Codeigniter, yaitu kerangka kerja yang menggunakan konsep M-V-C (*Model-View-Controller*) yang memungkinkan pemisahan antara *layer application-logic* dan *presentation* [8]. Nginx digunakan sebagai *web server* untuk menampung aplikasi agar dapat diakses. Nginx merupakan *web server* dan *reverse proxy* berkinerja tinggi yang didesain untuk menggunakan hanya sedikit sumber daya sistem [9]. Nginx menggunakan fitur mekanisme penanganan koneksi *event-based*, yaitu fitur yang mampu meminimalkan *thread* untuk memproses sebuah permintaan dari klien, yang mengakibatkan memori yang terpakai juga menjadi lebih kecil [10].

Konfigurasi pada *server* dilakukan secara *remote* menggunakan aplikasi Putty. Aplikasi Google Chrome sebagai penjelajah *web* yang berjalan pada sistem operasi Windows 10 x64 digunakan untuk melakukan percobaan dan pengujian berjalannya aplikasi.

D. Desain Topologi

Desain topologi sistem manajemen *log* terdiri dari berbagai entitas yang terlibat dalam pembangunan sistem. Entitas tersebut dihubungkan menggunakan suatu topologi star dengan switch sebagai pusat. Gambar 1 menunjukkan topologi fisik dari sistem manajemen *log*.

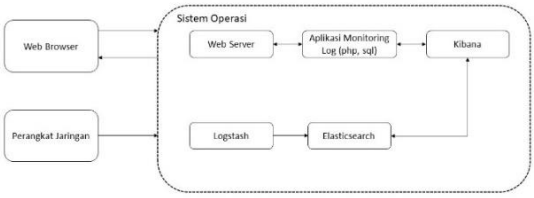


Gambar 1. Desain Topologi Sistem

Setiap perangkat memiliki fungsi masing-masing sebagai *manager*, *agent*, dan *user* yang menggunakan aplikasi sistem manajemen *log*.

E. Desain Sistem

Desain sistem memberikan gambaran per blok atau bagian sistem monitoring. Gambar 2 menunjukkan desain sistem *server*.



Gambar 2. Desain Sistem *Server*

F. Perancangan Proses Kerja

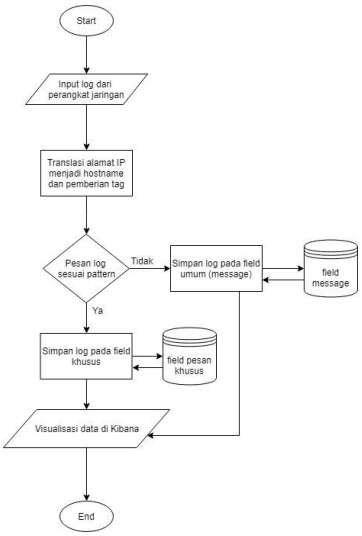
Perancangan dilakukan pada protokol *syslog* sebagai pengumpul informasi. Logstash sebagai salah satu aplikasi *syslog* untuk mengumpulkan *log* seluruh agen yang mengirim *log* pada *server* manajemen *log*. Pada pengiriman *log* dapat menggunakan jaringan TCP ataupun UDP. Pada penelitian ini, peneliti menggunakan jaringan UDP pada *port* nomor 8514.

Pengumpulan *log* bersifat automatis sehingga tidak diperlukan permintaan data pada *agent*. *Log* dari agen akan dikirimkan ke *server* sehingga *log* dapat diolah oleh Logstash dan disimpan pada Elasticsearch. Pada *server* diberi

pengaturan berupa jarigan yang akan dilalui UDP atau TCP, tingkat *severity*, *facility*, *facility mnemonic*, dan pesan dari masing-masing *log* yang dikirim *agent*. Pada *agent* diberi pengaturan berupa jaringan yang akan digunakan UDP atau TCP, dan tujuan pengiriman *log* berupa alamat *IP* dari *server* manajemen *log*.

G. Alir Proses Manajemen *Log*

Diagram alir proses menjelaskan mengenai proses pengumpulan informasi menggunakan *Syslog*. Informasi yang dikumpulkan akan disimpan ke dalam basis data dan informasi digunakan untuk memantau *log* sesuai sumber serta nilai *severity* yang dimiliki. Proses kerja sistem manajemen *log* ditunjukkan pada Gambar 3.



Gambar 3. Diagram alir proses manajemen *log*

III. HASIL DAN PEMBAHASAN

Pengujian sistem dilakukan untuk memeriksa kinerja sistem yang diimplementasikan. Tujuan utama dari pengujian sistem adalah untuk memastikan bahwa komponen-komponen dari sistem telah berfungsi sesuai yang dibutuhkan.

A. Pengujian Syslog

Berdasarkan kebutuhan fungsional, sistem manajemen *log server* harus dapat mengumpulkan serta mengelola *log* perangkat jaringan. Protokol komunikasi menggunakan *syslog*. Pengumpulan serta pengelolaan yang dimaksud adalah menampung *log* dari perangkat *agent* yang terdapat *syslog* ke aplikasi Elasticsearch serta mengidentifikasi *log* yang mengandung pesan khusus dan mengelompokkan *log* tersebut sesuai kriteria tingkat *severity* yang dimiliki.

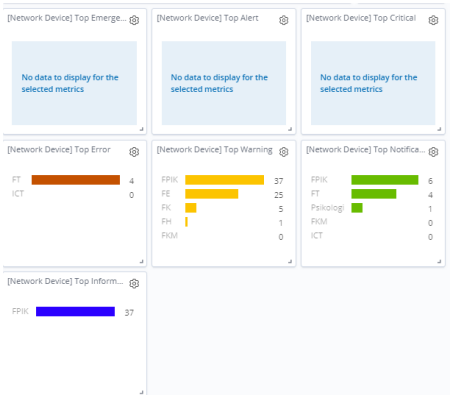
Pengujian dilakukan dengan mengamati halaman *Discover* pada Kibana sebagai *interface* untuk pengguna. Jika terdeteksi *log* yang dikirim, maka akan terlihat data yang masuk pada halaman *Discover* Kibana. Gambar 4 memperlihatkan hasil pengujian penerimaan *log*.

Gambar 4. *Log* diterima *server*

Gambar 4 memperlihatkan *log* dari perangkat jaringan berhasil diterima *server* serta dapat merepresentasikan identifikasi bagian-bagian pesan dari *log* yang diterima. Alamat *IP* dari perangkat yang mengirim *log* dapat ditranslasi menjadi *hostname* dan berhasil diberi *tag* unit berdasarkan *hostname* tersebut. Pesan-pesan khusus tersebut dapat divisualisasikan menggunakan *field* khusus untuk menyimpan pesan tersebut seperti yang ditunjukkan pada bagian *Available Fields* pada Gambar 4.

B. Pengujian *Severity Log*

Berdasarkan kebutuhan fungsional pada sistem manajemen *log*, dibutuhkan pengelompokkan *log* berdasarkan tingkat *severity* pada *log* tersebut. Pengelompokkan *log* memanfaatkan *field* severity_level pada Elasticsearch yang berguna sebagai parameter *filter* pemisahan *log*. Gambar 5 menunjukkan *log* yang sudah dikelompokkan berdasarkan tingkat *severity* yang dimiliki *log* tersebut.

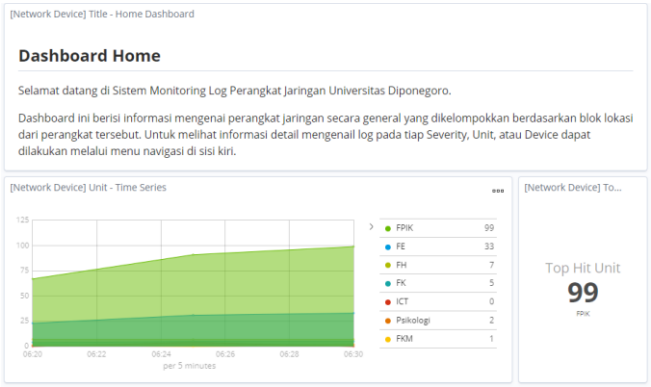


Gambar 5. Pengelompokan *log* berdasar tingkat *severity*

Berdasarkan Gambar 5 pengelompokkan *log* pada masing-masing *tag* unit berdasarkan tingkat *severity* masing-masing unit termasuk juga *top severity* per unit.

C. Pengujian *Dashboard Home*

Dashboard Home menampilkan informasi umum mengenai perangkat jaringan yang dikelompokkan berdasarkan unit lokasi penempatan perangkat tersebut. *Dashboard Home* terdiri dari visualisasi Unit *Time Series*, *Top Hit* Unit, *Severity Gauge*, dan *Top Severity*. Gambar 6 menunjukkan informasi-informasi yang terdapat pada *dashboard Home*.

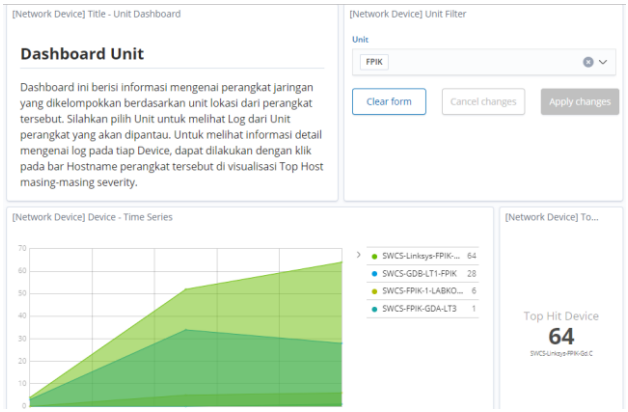


Gambar 6. Tampilan *dashboard Home*

Gambar 6 memperlihatkan hasil pengujian *dashboard Home* yang menunjukkan sistem dapat menerima *log* dari seluruh perangkat jaringan dan mengolah *log* tersebut menjadi informasi sesuai jenis visualisasi yang telah dikonfigurasi. Melalui *dashboard Home* dapat diketahui informasi mengenai unit yang paling banyak mengirim *log*, jenis *severity* yang paling banyak dilaporkan dan *time series* pengiriman *log* dari masing-masing unit perangkat jaringan.

D. Pengujian *Dashboard* Unit

Dashboard unit menampilkan informasi serupa seperti *dashboard Home*, tetapi *dashboard* unit menampilkan informasi lebih spesifik berdasarkan unit lokasi dari perangkat jaringan. Komposisi penyusun *dashboard* unit sama seperti *dashboard Home*. Gambar 7 menunjukkan hasil pengujian *dashboard* Unit.

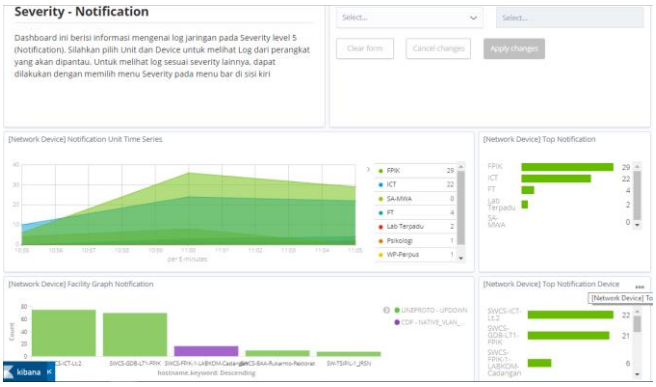


Gambar 7. Tampilan *dashboard* Unit

Gambar 7 memperlihatkan hasil pengujian *dashboard* unit menujukkan bahwa *dashboard* dapat menampilkan visualisasi *log* dari seluruh perangkat jaringan dengan pengelompokkan berdasarkan lokasi penempatan perangkat jaringan tersebut serta informasi perangkat-perangkat pada unit yang dikelompokkan berdasarkan tingkat *severity* yang dimiliki.

E. Pengujian *Dashboard Severity*

Dashboard Severity menampilkan informasi *log* perangkat jaringan pada tingkat *Severity* tertentu. Gambar 8 memperlihatkan hasil pengujian pada *dashboard severity level* 5 (*Notification*).

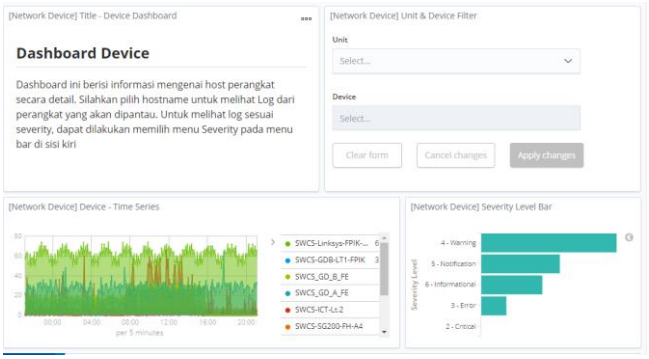


Gambar 8. Tampilan *dashboard Notification*

Gambar 8 memperlihatkan hasil pengujian *dashboard Notification* yang menunjukkan bahwa sistem dapat menampilkan visualisasi *log severity level* 5 (*Notification*) pada perangkat jaringan. Melalui *dashboard* juga dapat dilihat *time series* dari *log* dengan *severity notification*. Terdapat beberapa *facility* yang mendominasi *severity notification*, yaitu LINK, dan LINEPROTO. Dari Gambar 8 dapat diketahui juga unit yang paling banyak mengirim *log notification* adalah FPIK. Melalui visualisasi *dashboard Notification* juga grafik aktif dan tidak aktif suatu *interface*, pengguna yang mengakses perangkat, perangkat yang mengalami *threshold violation*, perangkat yang terhubung melalui protokol OSPF, perangkat yang di-*restart*, dan perangkat yang dapat diakses melalui SSH dapat diketahui.

F. Pengujian *Dashboard Device*

Dashboard Device menampilkan informasi *log* yang spesifik merujuk pada perangkat tertentu. Komposisi visualisasi penyusun *dashboard Device* terdiri dari visualisasi dari *log* yang sering dikirim oleh perangkat, diantaranya *Link Time Series*, *Native VLAN Mismatch*, *MAC Address Flapping*, dan *Duplex Mismatch*. Gambar 9 menunjukkan hasil pengujian *dashboard Device*.



Gambar 9. Tampilan *dashboard Device*

Gambar 9 memperlihatkan hasil pengujian *dashboard* Device yang menujukkan bahwa *dashboard* dapat menampilkan visualisasi *log* perangkat jaringan. Melalui *dashboard* juga dapat dipilih unit dan perangkat tertentu yang akan dipantau serta terdapat tabel *log* dan *log* yang paling sering dikirim perangkat untuk memudahkan analisa *log* dari suatu perangkat.

G. Pengujian Aplikasi

Bagian pengujian aplikasi menjelaskan hasil dan tampilan aplikasi yang telah dibuat. Tampilan awal pada aplikasi adalah halaman *login* yang berfungsi untuk autentikasi pengguna yang akan mengakses *server*.

Setelah pengguna terautentikasi, akan diarahkan ke halaman utama yang berisi *dashboard Home*. *Dashboard Home* menampilkan informasi umum dari perangkat jaringan. Terdapat beberapa menu navigasi yang tersedia pada aplikasi, seperti *System Overview*, *Severity*, Unit, dan *Device*. Masing-masing menu akan mengarahkan ke halaman *dashboard* sesuai dengan menu yang dipilih. Tampilan halaman utama diperlihatkan pada Gambar 10.

Gambar 10. Halaman utama aplikasi

IV. KESIMPULAN

Switch Cisco WS-C2960X-24TD-L dan Router Cisco 2921 sebagai agen pada perangkat jaringan mampu mengirimkan *log* ke server. *Server* manajemen *log* mampu mengumpulkan *log* dari perangkat pada jaringan. Pengumpulan *log* dari berbagai perangkat dengan menggunakan protokol UDP pada *port* 8514 dan aplikasi Logstash telah berhasil diimplementasikan. *Log* yang telah diterima berhasil dikelompokkan pada *field* Elasticsearch sesuai jenis pesan dari *log* tersebut. Informasi yang serupa telah terbukti berdasarkan hasil pengamatan memiliki tingkat *severity* yang berbeda, dengan perbandingan jumlah perangkat 1:4 tergantung versi sistem operasi dari perangkat jaringan. Tampilan pada *dashboard* Kibana dapat dibagikan dan ditampilkan pada aplikasi *web* lain menggunakan *iframe*.

Saran untuk penelitian lanjutan dapat dilakukan pengintegrasian data *log* dengan *email*, *sms*, atau aplikasi perpesanan untuk pemberitahuan kepada administrator. Penelitian lanjutan dapat mengelola *log server-server* aplikasi pada lingkungan ICT Universitas Diponegoro.

DAFTAR PUSTAKA

[1] K. . Ratnaningsih dan I. . Suaryana, "Pengaruh kecanggihan teknologi informasi, partisipasi manajemen, dan pengetahuan manajer akuntansi pada efektivitas sistem informasi akuntansi," *J. Akunt. Univ. Udayana*, vol. 6, no. 1, pp. 1–16, 2014.

[2] S. Wongkar dkk., "Analisa Implementasi Jaringan Internet Dengan Menggabungkan Jaringan LAN Dan WLAN Di Desa Kawangkoan Bawah Wilayah Amurang II," vol. 4, no. 6, pp. 62–68, 2015.

[3] A. Leskiw, "Understanding Syslog: *Server*s, Messages & Security," 2017. [Online]. Available: https://www.networkmanagementsoftware.com/what-is-syslog/. [Accessed: 14-Mar-2019].

[4] S. Alspaugh dkk., "Analyzing Log Analysis : An Empirical Study of User Log Mining This paper is included in the Proceedings of the Analyzing Log Analysis : An Empirical Study of User Log Mining user surveys," 2014.

[5] V. Anastopoulos dan S. Katsikas, "A structured methodology for deploying log management in WANs," *J. Inf. Secur. Appl.*, vol. 34, pp. 120–132, 2017.

[6] C. Preneur, "ELK (ElasticSearch , Logstash , Kibana)," 2018. [Online]. Available: https://medium.com/@adisaputra.id/elk-elasticsearch-logstash-kibana-a48c12612b16. [Accessed: 15-Mar-2018].

[7] S. Chaajed, *Learning ELK Stack*. Birmingham: Packt Publishing Ltd., 2015.

[8] Q. J. A. Mara Destiningrum, "Sistem Informasi Penjadwalan Dokter Berbassis Web Dengan Menggunakan Framework Codeigniter (Studi Kasus: Rumah Sakit Yukum Medical Centre)," *Teknoinfo*, vol. 11, no. 2, pp. 6–13, 2017.

[9] M. Hourani, Q. Shambour, A. Al-Zubidy, dan A. Al-Smadi, "Proposed Design and Implementation for RESTful Web *Server*," *J. Softw.*, vol. 9, no. 5, pp. 1071–1080, 2014.

[10] R. Sharma, *NGINX High Performance*. Mumbai: Packt Publishing Ltd, 2015.